ISAAC URBINA

MOBILE APPS COMPANY – ANDROID TRAINING

**WORKING WITH INSTANCE STATES**

Basically they are used to save a current state of the activity you are using, in case you want to restore it just as were you left, for example in videogames to save information about the last level that was open when you close the app or something.

You use 2 methods:

* onSaveInstanceState: It is used before the activity is no longer on the “Resumed” state (or running) and you can write extra information that you might want to use while opening the app the next time. It is useful within a session when moving between activities. However, it is important to mention that this is not persistent memory because this information will not be available if the App is destroyed. (The App, not the activity). This method is invoked before the onStop method of the App.
* onRestoreInstanceState: It is used to keep using a previous stored state of the activity, it is called just after the onStart method. This way you can retrieve the existing “session” that was open when you are coming back to this activity after either invoking a startIntent or startIntentforResult.

**CONTEXT**

The context is an interface containing the information of the App environment and it allows the App to use resources and classes of the App and the system. It is necessary to be able to launch activities, services, send and receive broadcasts and intents.

It is implicit in the activities, however, it is necessary to obtain it (but you can never create it) while using other things like AppWidgetProviders, Threads, etc. The methods used to invoke it are the following:

* getApplicationContext()
* getContext()
* getBaseContext()
* The reserved word *this* while inside any Activity.

**STARTACTIVITYFORRESULT**

This method is used to start a second activity, that can be part or not of your own application, and then get information or processing from the second activity back to the first activity. A request code needs to be sent while starting the activity, and when the second activity finishes what it has to do, it will send back the request code plus the result to the first activity. This way, multiple activities for result can be done and then you can recognize by the code what you were expecting.

The methods involved are:

* startActivityForResult(Intent, REQUEST\_CODE)
* onActivityResult(REQUEST\_CODE, RESULTCODE, data): The result code can be RESULT\_OK and RESULT\_CANCELED.

**PARCELABLE**

It is an interface that once instantiate, can write and restore Parcels. It is used to pass information between components between applications. To do this, you must create a Parcelable class and object, then this object is passed while starting an activity by adding it to the extras to the intent. Then it is obtained by getting the Intent and the extras from the savedInstanceState in the onCreate.

In the class that is implementing this interface, a static final CREATOR variable is needed in order for it to work.

**BACK STACK**

It is a pile of activities or applications stored in memory that are collected through the use of them. When pressing the back button, the user goes back to the previous activity that was open (even from another app). In order to manipulate it, we must use a TaskStackBuilder and change some flags.

In the manifest file we must add Launchmode and TaskAffinity, then to each intent we have to set:

* FLAG\_ATIVITY\_NEW\_TASK, to start an activity in a new task
* FLAG\_ACTIVITY\_CLEAR\_TASK, resumes the instance if it already exists and clears the back stack of the app.
* FLAG\_ACTIVITY\_SINGLE\_TOP, if the activity is already on top of the stack, it doesn’t create a new one. I didn’t get this one, but it is also not available in the official android documentation but forums.

To change the behavior of the back button in our apps, we must override the onBackPressed button and change the code there.

**SHAREDACTIONPROVIDER**

It is an android class that enables the creation of views to share any kind of data using applicatios that are enabled to receive this action. An xml file with the menu needs to be added to the project. It is mostly used to share links that are latter going to be open in apps configured to open those links, like facebook, twitter, pinterest, or send texts by SMS, WhatsApp, or save it to notes or something else that can handle text.